3 TOP 元件

3.1 介绍

图片纹理元件，即 TOP，几乎会在每个项目中用到。他们用于处理影片播放、3d 渲染、合成、硬件视频输入输出等这些动作时的图形操作；被用于表现任何输出到监视器、投影机、或 LED 上的东西。

3.2 Movie 元件

在 TOP 元件里，Movie 是最常用的种类之一。 它的功能是把资源加载进 TD。它能加载多种不同资源，从静态图片到各种视频编码格式。 下面是一部分 Movie 常用的文件格式。

.mov、.mp4、.avi、.jpg、.tiff、.png

还有很多其他支持的格式，在 wiki 的'File Types'页下有它们的清单.

在 Movie 中，有一些很棒的功能，可以大大减少那些令人头疼的不同帧速度的资源的采集和输出。一个主要的功能是，Movie 的目标是按真实时长播放资源 。 举个例子，如果项目设置是 60FPS，有一个 10S 的 30FPS 的资源，它会播够 10S 的时长，而不管项目和资源之间的帧速度的差异。相反也是一样。一个 60FPS 的 10S 资源，在一个 30FPS 的时间轴上播放，也一样会播放 10S。在上面这两种情况下，帧数被增加或减少，来匹配真实世界的时长。因此在某些情况下，使用插入帧会成为一个好办法。

3.3 预加载影片

当创建实时程序时，帧速度的下降会严重影响体验感受。后面的章节我们会讲到性能优化，除此之外，还有很多预防措施。

预先加载和卸载 Movie,就是一个办法。这个简单的步骤常常被新手忽略，因为这个最简单的办法需要脚本。

打开'Preloading.toe'的例子，这个例子里有三个按钮。
在'moviein1'元件的' 'Tune'参数下，有一个'Pre-Read frames'参数。
'Preload'按钮用下面的 Python 函数来预加载'Pre-Read frames'值所代表的帧数。

op('moviein1').preload()

'Play'按钮开始播放 Movie。'Unload'停止播放'moviein1'，卸载影片，释放系统资源。下面的 Python 脚本可以完成这些工作。

op('play').click(0)

op('moviein1').unload()

播放影片前，最好预加载，否则就等着播放的时候卡成狗吧。

3.4 Null TOP 和 Select TOP

相比那些费资源的 TOP,比如 Blur TOP，另外一些 TOP 是不需要占用系统资源的，能随便用。 比如 Null TOPs 和 Select TOPs。 尽管这两个 OP 不改变任何元素，但在创建高效的流程上非常有用。

一个布局合理的网络常被放在 Null TOP 和 Select TOP 里，而不是任由连线交错重叠，难以辨识。

打开例子'Null_1.toe'和 'Null_2.toe'。 第一个文件是一大堆 TOP 混在一起。这个项目没有考虑网络布局，接线被 OP 和其他接线重叠覆盖，难以判断 OP 之间的具体关联。

在第二个例子里，先用一些 Null TOP 汇集了所有信号，然后才将这些 TOP 组合起来。这些 Null TOP 可以作为一个节点，在快速浏览时，能更容易的追踪 OP 之间的联系

Select TOP 也是一样。当使用嵌套网络，使用 Out TOP，在容器之间连线，会产生和上面一样的情况，网络很快变得难以辨认。Select TOP 则可以快捷整齐地引用其他 TOP。 打开例子 1，这个例子演示了用 In TOP 和 Out TOP 会额外导致怎样的杂乱。这个例子也只是复制了 12 遍影片。回头咱要是需要复制 100 份咋办？这就是用 Select TOP 的方便之处。

打开例子 2，这个例子成倍增加了所复制的组件的数量，同时显得更清晰明了。更有趣的是用 Select TOP 创建的动态选择系统。这比之前的手动方法要高效的多，允许用 Select TOP 的 Select 参数中的 Python 脚本，根据名称，来自动引用从上面网络中复制来的相应 TOP。在这个观念上更进一步，在 Replicator 复制器的使用上，如果 master 中能有一个 selectDAT，每个新产生的子节点都会引用外部数据。如果觉得这里例子很难，不要担心，后面的例子会涉及复制和脚本。现在，最重要的是知道，通过使用 Select TOP 和简单的脚本，组件会变得相对可持续，而且便于维护。 当我们需要复制更多项目，会变得跟向表格中添加行一样容易。

3.5 编码器

视频播放是一个繁杂的过程。 明智的做法是花时间尝试不同的编解码器，看哪种在视觉效果和性能的均衡上最适合该项目。

在开始操作具体的编码器前，了解编码器和容器之间的区别很重要。 编码器是音视频文件合格的总称，它容易让初学者弄混。 因为容器可以容纳多种编码器。

编码器是用来压缩和解压的。它有两个主要任务，第一个是压缩视频数据以便于储存和传输；第二个是解压视频数据来播放。因为这两种不同任务的存在，每个编码器有不同的侧重。一些倾向于把文件压缩成小体积，便与传输；而另一些侧重压缩为画质高，适合长期保存的影响。不同项目有不同需求，有时候，目标是用最高质量播放一个内容，另一些时候，必须降低画质来同时播放多个文件。为项目选择正确的编码器需要经过一些测试和思考，但可能会花不少时间。

容器的作用，跟它的名字一致。它可以容纳压缩的视频，音频，以及所有一个影片需要解压个播放的数据内容。在 TD 中有多种不同的容器，不过相对于编码器，他们对整个项目的流程影响区别不大。

当不同种类的容器和编码器组合使用，事情会变得复杂。想像下，有个叫 'test_movie.mov' 的视频文件。在一个项目中，这个文件是个 QuickTime 容器中的一段 Animation 格式压缩编码的.mov 文件。有趣和让初学者迷茫的是，在另一个项目里，它是一段 H.264 压缩编码的文件。更混乱的是，这个 H.264 文件可能还在一个 MPEG-4 容器中，用 '.mp4'做后缀。

抛开这些混乱，目前的 HAP 家族中用很多流行的编码器可选。H.264, Animation 编码, 还有 Cineform。每个编解码器都有自己的优点和缺点。下面是这些编码器的优缺点对比。

HAP 家族：
Pros
优点：
可以播放极高分辨率和高帧速率视频
极低的 CPU 消耗
HAP Q 是视觉无损压缩
极低的 GPU 消耗

缺点
大文件大小
难在窗口上编码文件
必须使用固态硬盘或 SSD RAID0 文件播放
主要瓶颈是硬盘读取速度

H.264

优点：
可制作轻量级视频
视频画质相对压缩程度，做的最好。
磁盘使用率低

缺点
需要大量的处理器内核来播放极高的分辨率或高帧速率。
如果在编码中不正确的话，会感到颜色分层化
比特率随内容变化明显
长宽分辨率都是 409
难以创建 alpha 通道

Animation Codec
优点

100%无损文件
优先考虑质量
自带支持 Alpha 通道

缺点：
文件很大
对硬盘和 CPU 的要求都很高
比特率随内容变化明显

Cineform
优点

恒定比特率
高图像质量
自带支持 Alpha 通道

缺点：
文件大
必须购买 cineform 软件编码